Slay Some Earthly Demons VI
Document: n3344
Author: Martin Uecker
Date: 2024-09-08

This remaining issues from n3244 were split up into the following series:

n3340 - Slay Some Earthly Demons II
n3341 - Slay Some Earthly Demons III
n3342 - Slay Some Earthly Demons IV
n3343 - Slay Some Earthly Demons V
n3344 - Slay Some Earthly Demons VI
n3345 - Slay Some Earthly Demons VII
n3346 - Slay Some Earthly Demons VIII
n3347 - Slay Some Earthly Demons IX

Meeting meeting notes about previous discussion can be found in n3281. The following numbers correspond to N3220 and the ones with (→) to N3301.

	J.2.19 needs more work, cf. SC22WG14.25600 (**note to editor:** parts applied by mistake)
	J.2.21 done (incl. additional change), (**note to editor:** there is a “not“ missing in 6.3.2.1)
n3340	J.2.40 (→ J.2.38) withdrawn in June, now revised (cf. SC22WG14.25600)
	J.2.56 done
	J.2.57 done (option 1)
n3341	J.2.58 (→ J.2.54) not discussed in June
	J.2.60 (→ J.2.56) direction to editor (**note to editor:** not yet done)
n3342	J.2.63 (→ J.2.59) no consensus in June, now revised to make implementation-defined
n3343	wording change not related to an entry in Annex J
	J.2.67 done
	J.2.69 done (alternative wording for semantic section)
n3344	**J.2.75 (→ J.2.70) now revised (cf. SC22WG14.25600)**
	J.2.77 (→ J.3.13. 2) not discussed in June, but fixed editorially
	J.2.78 (→ J.3.13. 3) not discussed in June, but fixed editorially
n3345	J.2.79 (→ J.2.72) not discussed in June
n3346	J.2.80-J.2.82 (→ J.2.73-J.2.75) not discussed in June (cf. SC22WG14.25600)
n3347	J.2.87 (→ J.2.80) not discussed in June

Acknowledgments: Thanks to Joseph Myers, David Svoboda, Chris Bazley, and the UB study group for helpful comments and corrections.

Annex (J.2.70 N3301, J.2.75 in N3220)

(70) A storage-class specifier or type qualifier modifies the keyword void as a function parameter type list (6.7.7.4).

Example

https://godbolt.org/z/cfPcvr8cs

```
int f(static void);
int g(register void);
int h(volatile void);
int i(const void);

int j(void x);
int j(void x) { }
```

Analysis **(updated, cf SC22WG14.25600)**

GCC rejects all, except the declaration with plain **void** and names parameter which only causes a warning. Clang allows the case with register. MSVC accepts register/volatile/const but rejects plain void with named parameter. Note that storage-class specifier other than **register** are already disallowed for parameters. Declaring a named parameter of type **void** seems to be allowed in a function declaration (but is rejected by Clang and MSVC), but can then not be used for a function definition because it is an incomplete type.

One option is to only allow a plain **void**, which is the only case that makes sense. This case corresponds to the special case of no parameters which makes it distinct from all other use cases for **void**. On the other hand, it is also not clear how allowing register and qualifiers or a parameter name would cause undefined behavior or other problems, as they are all ignored in function declarations. So one could simply remove this item from annex J.2. and insist that such all declarations are valid.

Proposed Change, Alternative 1

6.7.7.4 Function declarators

Constraints

4 A parameter declaration shall not specify a void type, except for the special case of a single unnamed parameter of type void with no storage class specifier, no type qualifier, and no following ellipsis terminator.

Proposed Change, Alternative 2

6.7.7.4 Function declarators

17 EXAMPLE 3 Valid but useless function declarations.

```
int g(register void);
int h(volatile void);
int i(const void x);
```